Group 6

Final Presentation

1061443 李杰穎
1061416 許巧臻
1063337 張仲緯

TABLE OF CONTENTS

01

Introduction

Introduction

- Kaggle: Predict Future Sales
  https://www.kaggle.com/c/competitive-data-science-predict-future-sales/
- Predict total sales for every product and store in the next month.

Motivation

Hope to become a “Sales Predictor”

02

Related Work

Related Work

1. Predicting Future Sales of Retail Products using Machine Learning [2020-08-18]
2. Sales Demand Forecast in E-commerce using a Long Short-Term Memory Neural Network Methodology [2019-01-13]

03

Data collection

Source

- Kaggle: Predict Future Sales

  https://www.kaggle.com/c/competitive-data-science-predict-future-sales/

Data description

- **sales_train.csv** - the training set. Daily historical data from January 2013 to October 2015.
- **test.csv** - the test set. You need to forecast the sales for these shops and products for November 2015.
- **sample_submission.csv** - a sample submission file in the correct format.
- **items.csv** - supplemental information about the items/products.
- **item_categories.csv** - supplemental information about the items categories.
- **shops.csv** - supplemental information about the shops.

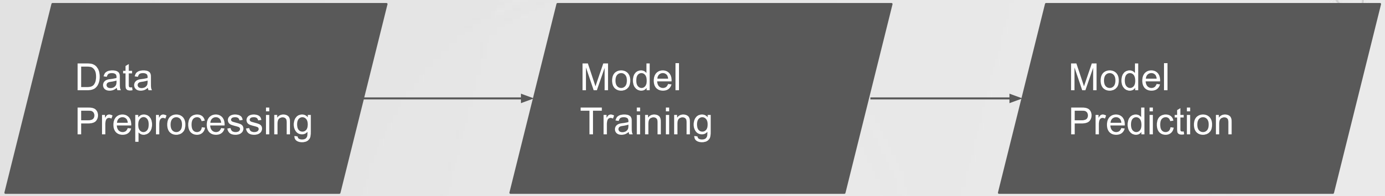
Data fields

- **ID** - an Id that represents a (Shop, Item) tuple within the test set
- **shop_id** - unique identifier of a shop
- **item_id** - unique identifier of a product
- **item_category_id** - unique identifier of item category
- **item_cnt_day** - number of products sold. You are predicting a monthly amount of this measure
- **item_price** - current price of an item
- **date** - date in format dd/mm/yyyy
- **date_block_num** - a consecutive month number, used for convenience. January 2013 is 0, February 2013
  - is 1,..., October 2015 is 33
- **item_name** - name of item
- **shop_name** - name of shop
- **item_category_name** - name of item category

04

Methodology

Flow diagram



Methodology

- LSTM
- RandomForest、XGBoost

Methodology

- LSTM
- RandomForest、XGBoost

Data preprocessing

- sales_train.csv date field string format convert to datetime format
- Create pivot table from sales_train.csv

	shop_id	item_id	item_cnt_day																		
date_block_num			0	1	2	3	4	5	6	7	...	24	25	26	27	28	29	30	31	32	33
0	0	30	0	31	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1	0	31	0	11	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2	0	32	6	10	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	0	33	3	3	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4	0	35	1	14	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

- `dataset = pd.merge(test_data,dataset,on = ['item_id','shop_id'],how = 'left')`

Data preprocessing

- Fill NaN values with 0
- dataset.drop(['shop_id','item_id','ID'],inplace = True, axis = 1)

	(item_cnt_day, 0)	(item_cnt_day, 1)	(item_cnt_day, 2)	(item_cnt_day, 3)	(item_cnt_day, 4)	(item_cnt_day, 5)	(item_cnt_day, 6)	(item_cnt_day 7)
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Data preprocessing

- Train set: 0 ~ 32 months, Label: 33 month
- Test set: 1 ~ 33 months

Model structure

- 1 LSTM layer
- 1 Dropout layer
- 1 Dense layer
- Loss: mean squared error
- Optimizer: adam
- Batch_size: 4096
- Epochs: 10

Layer (type)	Output Shape	Param #
lstm_1 (LSTM)	(None, 64)	16896
dropout_1 (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 1)	65

Total params: 16,961

Trainable params: 16,961

Non-trainable params: 0

Results

- Loss = mse, Batch_size = 4096, epochs = 10, Public score: 1.02098
- **Loss = rmse, Batch_size = 4096, epochs = 10, Public score: 1.01950**
- Loss = rmse, Batch_size = 4096, epochs = 20, Public score: 1.02579
- Loss = rmse, Batch_size = 4096, epochs = 100, Public score: 1.02515
- Loss = rmse, Batch_size = 4096, epochs = 1000, Public score: 1.02914
- Loss = rmse, Batch_size = 512, epochs = 200, Public score: 1.02549
- Loss = rmse, Batch_size = 128, epochs = 10, Public score: 1.02463
- Loss = rmse, Batch_size = 2048, epochs = 100, Public score: 1.02806

Methodology

- LSTM
- RandomForest、XGBoost

Data preprocessing

column	describe
date_block_num	a consecutive month number
shop_id	unique identifier of a shop
item_id	unique identifier of a product
item_category_id	unique identifier of item category
item_cnt_month	number of products sold
city_code	each shop_name starts with the city name
type_code	each category contains type and subtype in its name
subtype_code	each category contains type and subtype in its name
month	month of date
days	day of date
item_shop_last_sale	months since the last sale for each shop/item pair only
item_last_sale	months since the last sale for each item only
item_shop_first_sale	months since the first sale for each shop/item pair only
item_first_sale	months since the first sale for each item only

PC - Гарнитуры/Наушники

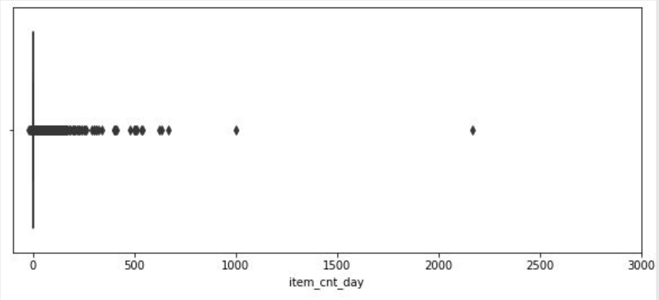
Data preprocessing

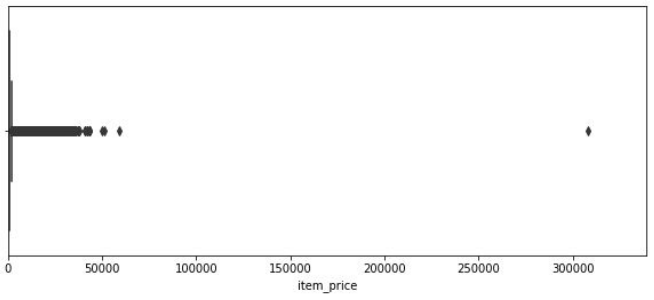
column	describe
item_cnt_month_lag_[i]	`i` monthly sales of each item
date_avg_item_cnt_lag_[i]	`i` average monthly sales of each item `groupby(date_block_num)`
date_item_avg_item_cnt_lag_[i]	`groupby(date_block_num, item_id)`
date_shop_avg_item_cnt_lag_[i]	`groupby(date_block_num, shop_id)`
date_cat_avg_item_cnt_lag_[i]	`groupby(date_block_num, item_category_id)`
date_shop_cat_avg_item_cnt_lag_[i]	`groupby(date_block_num, shop_id, item_category_id)`
date_city_avg_item_cnt_lag_[i]	`groupby(date_block_num, city_code)`
date_item_city_avg_item_cnt_lag_[i]	`groupby(date_block_num, item_id, city_code)`

Mean encoded features

Data preprocessing

- remove items with price > 100000 and sales > 1001





Data preprocessing

- Train set: 1 ~ 32 months
- Valid set: 33 month
- Test set: 34 month

Hyperparameter optimization

- Optuna is an automatic hyperparameter optimization software framework, particularly designed for machine learning.

OPTUNA

XGBoost parameter and Result

- max_depth=8
- n_estimators=1000
- min_child_weight=300
- colsample_bytree=0.8
- subsample=0.8
- eta=0.3
- seed=42
- **Kaggle score=0.92613**

XGBoost parameter and Result

- max_depth=12
- n_estimators=300
- min_child_weight=162
- colsample_bytree=0.6
- subsample=0.8
- eta=0.008
- seed=42
- **Kaggle score=0.91077**

- **without lag feature**
- max_depth=12
- n_estimators=300
- min_child_weight=162
- colsample_bytree=0.6
- subsample=0.8
- eta=0.008
- seed=42
- **Kaggle score=1.02610**

XGBoost parameter and Result

- max_depth=13
- n_estimators=3000
- min_child_weight=175
- colsample_bytree=0.5
- subsample=0.6
- eta=0.008
- seed=42
- **Kaggle score=0.89647**

RandomForest parameter and Result

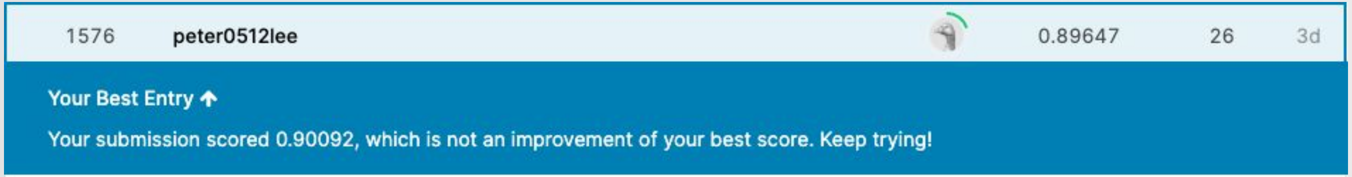
- max_depth=8
- n_estimators=70
- random_state=0
- n_jobs=-1
- **Kaggle score=0.93215**

- **without lag feature**
- max_depth=8
- n_estimators=70
- random_state=0
- n_jobs=-1
- **Kaggle score=1.09043**

05
Results

Results

- **Best Score Rank:** 1576 / 11877 (13%)



	Kaggle Score		
LSTM	1.01950		
XGBoost	0.89647	1.02610 (w/o lag feature)	0.92784 (paper)
RandomForest	0.94685	0.93215 (w/o lag feature)	

06

Conclusion

Conclusion

- XGBoost is a great choice
- We think LSTM can be more better
- Lag feature is necessary

Demo

ML_Demo.mov